HW 5
CS 6083 sections A and INET, Fall 2024, Prof Frankl
PROBLEM 1:

Consider the following B+ tree. The structure is described first, followed by the key values in each node. **n = 8**. (Therefore internal nodes other than the root must have between 4 and 8 children and leaves must have between 4 and 7 key values.)

Structure:
The root is node R.
Node R has two children: A and B.
A has four children: C, D, E, F.
B has eight children: G, H, I, J, K, L, M, N.

Internal node Key and pointer values:
R: p_A 400 p_B
A: p_C 100 p_D 180 p_E 320 p_F
B: p_G 480 p_H 620 p_I 740 p_J 820 p_K 1000 p_L 2000 p_M 3000 p_N
Leaf nodes with key values (and pointers to data file, not shown)
C: 20, 40, 60, 80
D: 100, 120, 140, 160
E: 180, 200, 220, 240, 260, 280, 300
F: 320, 340, 360, 380
G: 400, 420, 440, 460
H: 480, 500, 520, 540, 560, 580, 600
I: 620, 640, 660, 680, 700, 720
J: 740, 760, 780, 800
K: 820, 840, 860, 880
L, M, N: details not needed for this problem

1. Assume that the attribute being indexed has a UNIQUE constraint, so there is at most one record with each value for this attribute in the data file . Describe which nodes are visited, what key comparisons are made in each, and what data is fetched from the data file, when doing each of the following searches:

    a. find the record with key value 680
    b. find the record with key value 620
    c. find all records with key values between 685 and 825, inclusive,assuming the B+ tree is a sparse primary (clustering) index on the data file.
    d. find all records with key values between 680 and 820, inclusive,assuming the B+ tree is a secondary (non-clustering) index on the data file.

2. Describe the tree after each of the following operations. Please highlight the changes (or only describe/show the changes) to make it easier for the grader to check your work. For example, if you're splitting node F, you could say "split node F into F and F'. Node F has keys ... and F' has keys ..." **You must use the algorithms presented in the textbook / lecture 10**

*DO EACH OF THESE **SEPARATELY,** STARTING WITH THE ORIGINAL TREE DESCRIBED ABOVE:*

a. insert 795
b. insert 192
c. insert 593
d. delete 200
e. delete 340
f. delete 40

Problem 2

Consider this relational schema from the University database:

Takes(year, sem, courseID, secID, sID, grade)

Assume that the Block size = 4K **and consider** B+ trees where each tree node fits into a block RIDs (pointers to tree nodes and to records on disk) are 8 bytes. Assume the sID attribute is also 8 bytes.
There are 500,000,000 takes records, each 64B.

Consider using a B+ tree to build a secondary (non-clustering) index on *Takes.sID* . Show your work and state any assumptions you're making in answering the following:

A. Recall that a B+ tree has a parameter *n* that determines the min and max number of key values and pointers in each node. Determine the largest value of *n* that allows *n* pointers and *n-1* keys to fit into a block.
B. Choose a convenient number *x* to work with, somewhere between n/2 and n, assume each node has *x* children, and estimate the height of the tree. Note that there are many records for each *sID;* assume that there are duplicates of the same *sID* in the leaf nodes, so that the number of key values is equal to the number of records in *Takes*.
C. How many block transfers and how many seeks are needed to use this B+ tree to find all of the *Takes* records for the student with ID 12345, assuming that this student has taken 10 courses. Note how many of these are for searching the tree and how many are for finding records in the *Takes* file.
D. What would differ if the tree considered in A, B, C were a *primary (clustering)* index?
    a. What conditions would the data have to satisfy?
    b. Would the height of the tree be the same or different?
    c. What would differ in the procedure for and cost of the search for student 12345's *Takes* records?